100X: Leveraging the Future Potential of US Exascale Computing Project Investments

EXASCALE COMPUTING PROJECT

Michael A. Heroux, Sandia National Laboratories
Director of Software Technology

Advanced Modeling & Simulation (AMS) Seminar Series

NASA Ames Research Center, June 20th, 2023

National Nuclear Security Administration

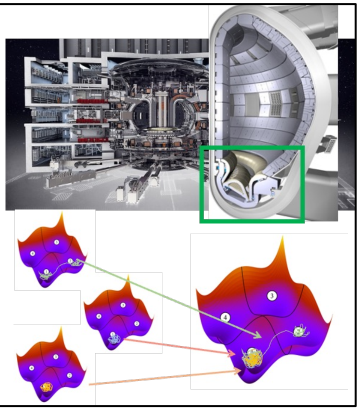
U.S. DEPARTMENT OF ENERGY | Office of Science

Outline

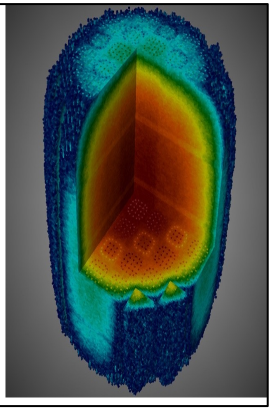
- 100X, Illustrated
- ECP, Briefly
- ECP Libraries and Tools, A Sample of Products
- Developing software for GPU systems
- Establishing software ecosystems
- 100X Opportunities and Recipes

100X Demonstrated: ECP-sponsored application FOMs

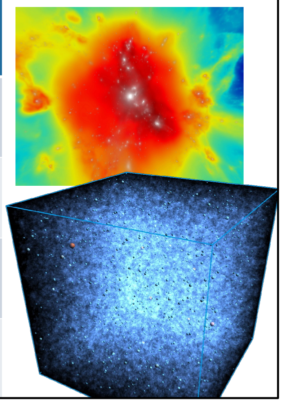
Project/PI	**EXAALT: Molecular Dynamics** **Danny Perez**
Challenge Problem	Damaged surface of Tungsten in conditions relevant to plasma facing materials in fusion reactors • 100,000 atoms • T=1200K
FOM Speedup	**398.5**
Nodes Used	**7000**
ST/CD Tools	Used in KPP Demo: **Kokkos, CoPa**



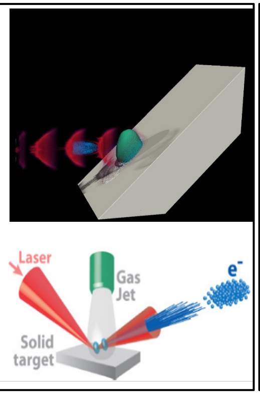
Project/PI	**ExaSMR: Small Modular Reactors** **Steve Hamilton**
Challenge Problem	NuScale-style Small Module Reactor (SMR) with depleted fuel and natural circulation • 213,860 Monte Carlo tally cells/6 reactions • 5.12×10^{12} particle histories/cycle, 40 cycles • 1098×10^{6} CFD spatial elements • 376×10^{9} CFD degrees of freedom • 1500 CFD timesteps
FOM Speedup	**70**
Nodes Used	**6400**
ST/CD Tools	Used in KPP Demo: **CEED** Additional: Trilinos



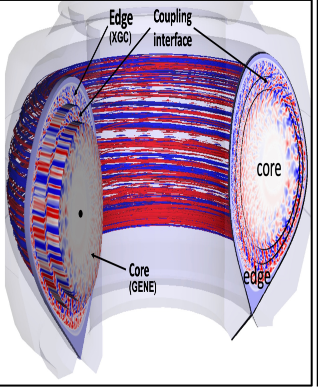
Project/PI	**ExaSky: Cosmology** **Salman Habib**
Challenge Problem	Two large cosmology simulations • gravity-only • hydrodynamics
FOM Speedup	**271.65**
Nodes Used	**8192**
ST/CD Tools	Used in KPP demo: **none** Additional: CoPa, VTK-m, CINEMA, HDF5.0



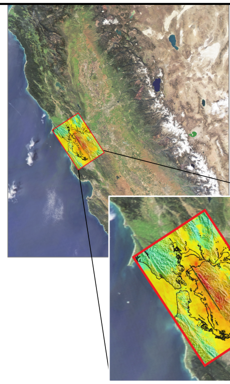
Project/PI	**WarpX: Plasma Wakefield Accelerators** **Jean-Luc Vay**
Challenge Problem	Wakefield plasma accelerator with a 1PW laser drive • 6.9×10^{12} grid cells • 14×10^{12} macroparticles • 1000 timesteps/1 stage
FOM Speedup	**500**
Nodes Used	**8576**
ST/CD Tools	Used in KPP Demo: **AMReX**, **libEnsemble** Additional: ADIOS, HDF5, VTK-m, ALPINE



Project/PI	**WDMApp: Fusion Tokamaks** **Amitava Bhatacharjee**
Challenge Problem	Gyrokinetic simulation of the full ITER plasma to predict the height and width of the edge pedestal
FOM Speedup	**150**
Nodes Used	**6156**
ST/CD Tools	Used in KPP Demo: **CODAR**, **CoPA**, **PETSc**, **ADIOS** Additional: VTK-m



Project/PI	**EQSIM: Earthquake Modeling and Risk** **Dave McCallen**
Challenge Problem	Impacts of Mag 7 rupture on the Hayward Fault on the bay area.
FOM Speedup	**3467**
Nodes Used	**5088**
ST/CD Tools	Used in KPP Demo: **RAJA, HDF5**

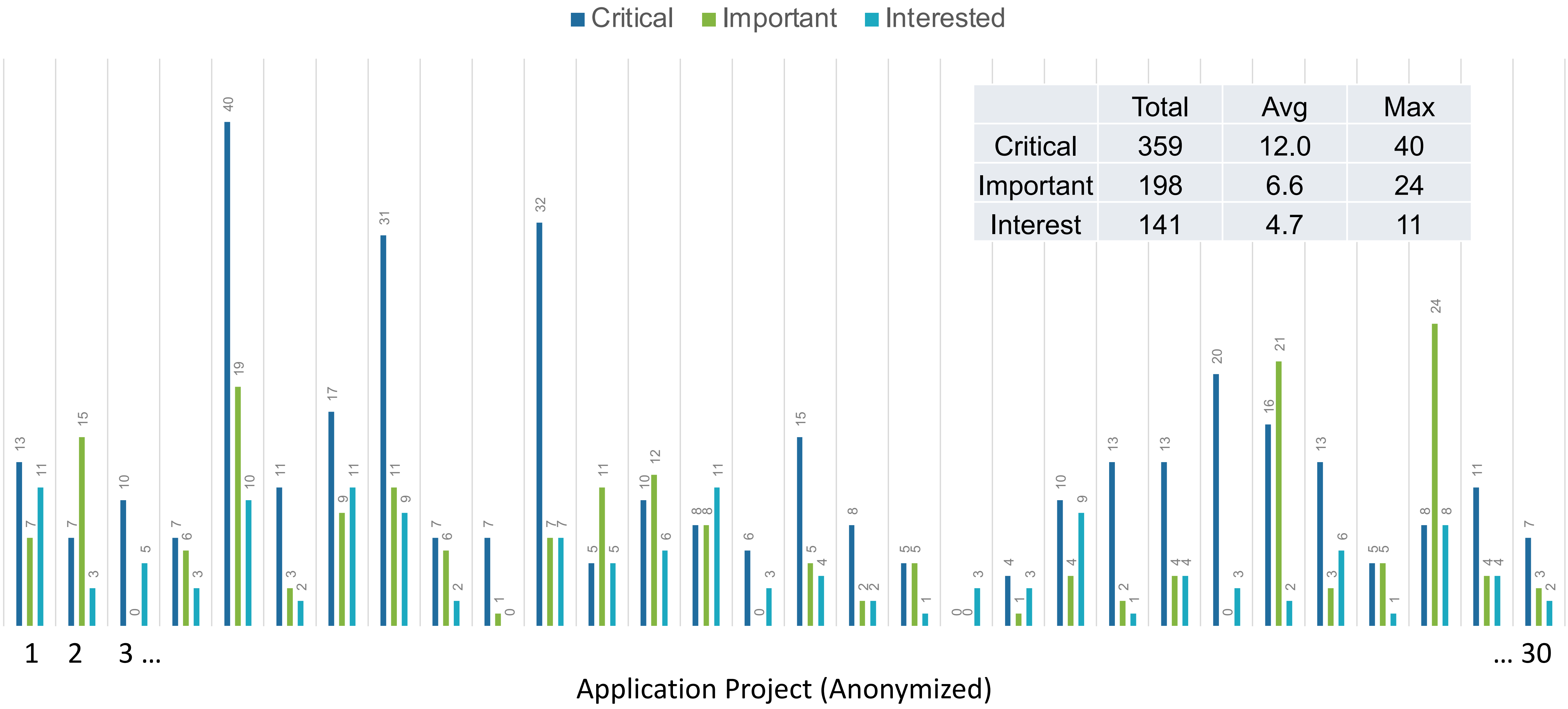


EXASCALE COMPUTING PROJECT

ECP's KPPs: Quantified with Explicit Targets

KPP ID	Description of Scope	Threshold KPP	Objective KPP	Verification Action/Evidence
KPP-1	11 selected applications demonstrate performance improvement for mission-critical problems	✓ 6 of 11 applications demonstrate Figure of Merit improvement ≥50 on their base challenge problem	All 11 selected applications demonstrate their stretch challenge problem	Independent assessment of measured FOM results and base challenge problem demonstration evidence
KPP-2	14 selected applications broaden the reach of exascale science and mission capability	5 of 10 DOE Science and Applied Energy applications *and* 2 of 4 NNSA applications demonstrate their base challenge problem	All 14 selected applications demonstrate their stretch challenge problem	Independent assessment of base challenge problem demonstration evidence
KPP-3	76 software products selected to meet an aggregate capability integration score	Software products achieve an aggregate capability integration score of at least 34 out of a possible score of 68 points	Software products achieve the maximum aggregate capability integration score of 68 points	Independent assessment of each software product's capability integration score
KPP-4	Delivery of 267 vendor baselined milestones in the PathForward element	✓ Vendors meet 214 out of the total possible 267 PathForward milestones	✓ Vendors meet all 267 possible PathForward milestones	Independent review of the PathForward milestones to assure they meet the contract requirements; evidence is the final milestone deliverable

THE NUMBER OF ECP SOFTWARE TECHNOLOGY PROJECT DEPENDENCIES FOR EACH ECP APPLICATION PROJECT (ANONYMIZED)



	Total	Avg	Max
Critical	359	12.0	40
Important	198	6.6	24
Interest	141	4.7	11

EXASCALE COMPUTING PROJECT

ECP investments enabled a 100X improvement in capabilities

- **7 years** building an **accelerated**, **cloud-ready** software ecosystem
- Positioned to utilize **accelerators from multiple vendors** that others cannot
- **Emphasized software quality**: testing, documentation, design, and more
- Prioritized **community engagement**: Webinars, BOFs, tutorials, and more
- DOE **portability layers** are the credible way to
  - Build codes that are sustainable **across multiple GPUs** and
  - **Avoid vendor lock-in**
  - **Avoid growing divergence** and hand tuning in your code base
- ECP software can **lower costs** and **increase performance** for **accelerated** platforms
- Outside of AI, industry has not caught up
  - DOE enables an entirely different class of applications and capabilities to use accelerated nodes
  - In addition to AI
- **ECP legacy: A path and software foundation for others to leverage**

ECP in a Nutshell

ECP's holistic approach uses co-design and integration to achieve exascale computing

Performant mission and science applications at scale

Aggressive RD&D project	Mission apps; integrated S/W stack	Deployment to DOE HPC Facilities	Hardware technology advances

Application Development (AD)

Develop and enhance the predictive capability of applications critical to DOE

25 applications
National security, energy, Earth systems, economic security, materials, data

6 Co-Design Centers
Machine learning, graph analytics, mesh refinement, PDE discretization, particles, online data analytics

Software Technology (ST)

Deliver expanded and vertically integrated software stack to achieve full potential of exascale computing

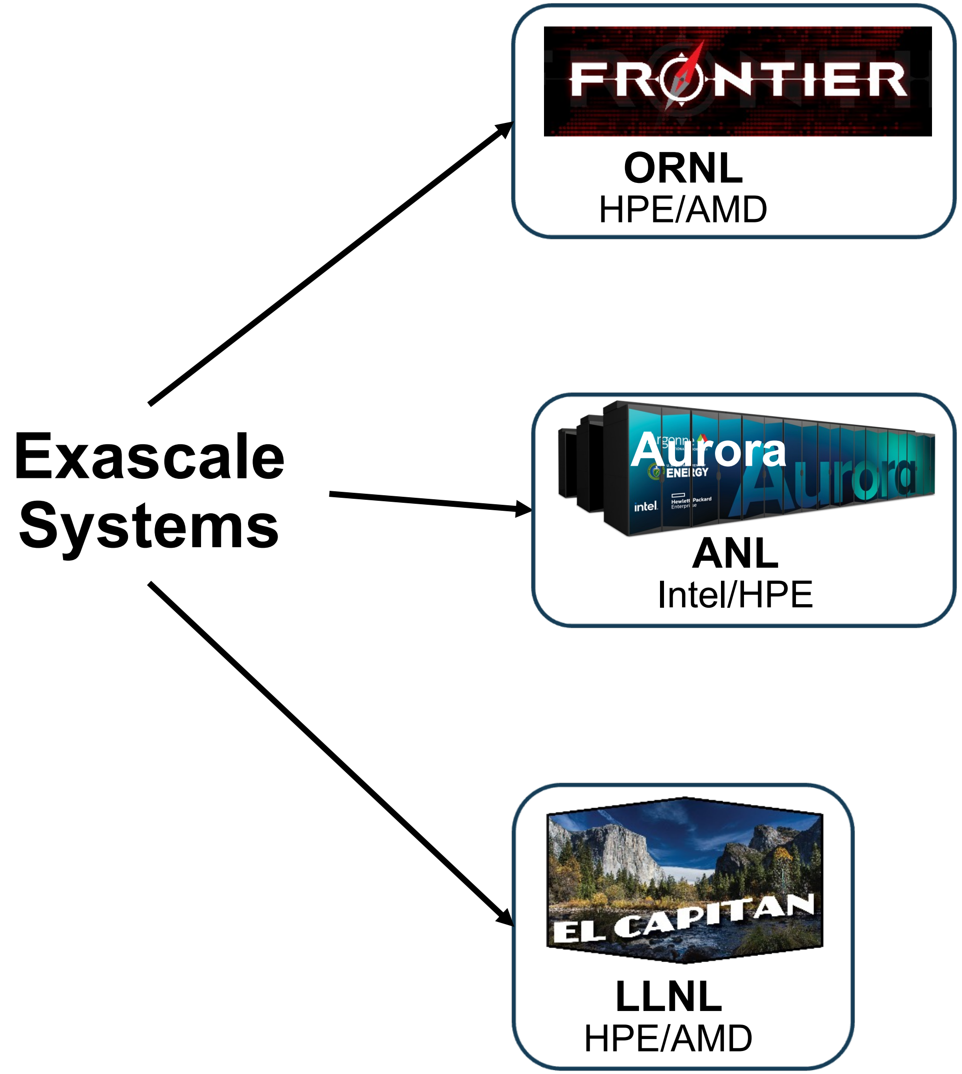
70 unique software products
spanning programming models and run times,
math libraries,
data and visualization

Hardware and Integration (HI)

Integrated delivery of ECP products on targeted systems at leading DOE HPC facilities

6 US HPC vendors
focused on exascale node and system design; application integration and software deployment to Facilities

Exascale Systems – Primary targets for ECP Software Teams



- ECP libraries & tools migrating to GPU platforms
- Target AMD, Intel and Nvidia (Perlmutter) devices
- Growing support for Arm/SVE in the same stack
- Mature MPI/CPU stack also robust and evolving
- Eye toward specialized devices, e.g., dataflow
- Legacy:
  - A stack to support application portability
  - Across many different distributed systems with
  - Multiple kinds of devices (GPUs, CPUs, etc)

ECP Software Technology works on products that apps need now and in the future

Key themes:
- Focus: GPU node architectures and advanced memory & storage technologies
- Create: New high-concurrency, latency tolerant algorithms
- Develop: New portable (Nvidia, Intel, AMD GPUs) software product
- Enable: Access and use via standard APIs

Software categories:
- **Next generation established products:** Widely used HPC products (e.g., MPICH, OpenMPI, PETSc)
- **Robust emerging products:** Address key new requirements (e.g., Kokkos, RAJA, Spack)
- **New products:** Enable exploration of emerging HPC requirements (e.g., SICM, zfp, UnifyCR)

Legacy: A stack that enables performance portable application development on leadership platforms

Example Products	Engagement
MPI – Backbone of HPC apps	Explore/develop MPICH and OpenMPI new features & standards
OpenMP/OpenACC –On-node parallelism	Explore/develop new features and standards
Performance Portability Libraries	Lightweight APIs for compile-time polymorphisms
LLVM/Vendor compilers	Injecting HPC features, testing/feedback to vendors
Perf Tools - PAPI, TAU, HPCToolkit	Explore/develop new features
Math Libraries: BLAS, sparse solvers, etc.	Scalable algorithms and software, critical enabling technologies
IO: HDF5, MPI-IO, ADIOS	Standard and next-gen IO, leveraging non-volatile storage
Viz/Data Analysis	ParaView-related product development, node concurrency

ECP Libraries and Tools

A Sampler of Products

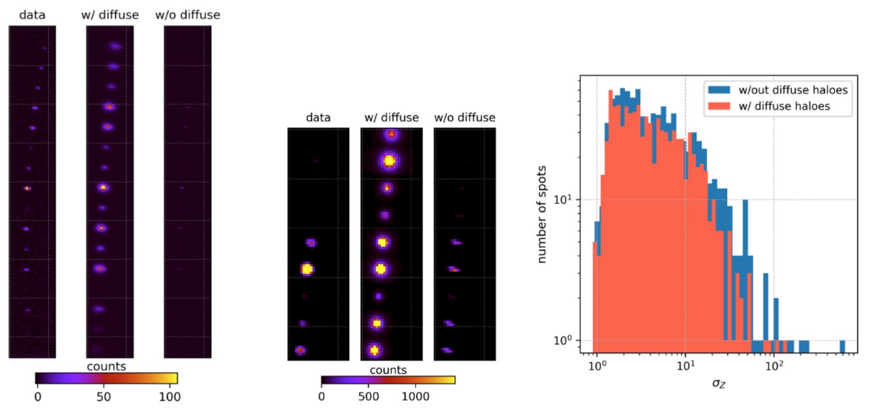
- No two project alike
- Some personality driven
- Some community driven
- Small, medium, large

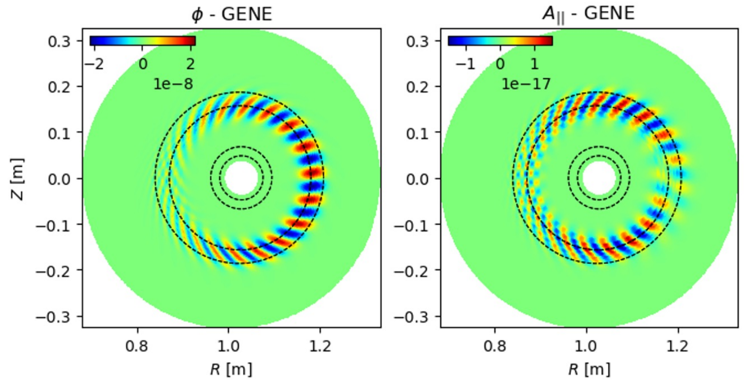
Integration: AD Teams Depend Heavily on ST Software to Meet KPPs

Kokkos

ExaFEL

nanoBragg code ported from Nvidia to AMD GPUs with minimal effort





ADIOS

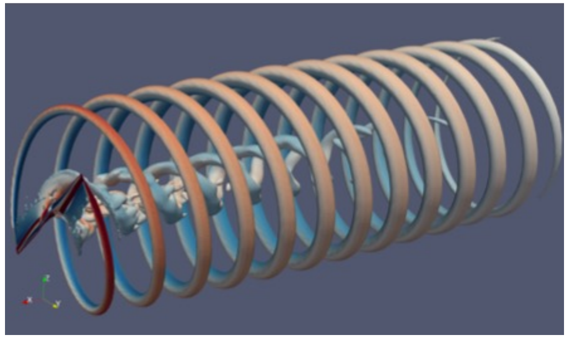
WDMApp

ADIOS enables in-memory coupling between GENE and XGC

hypre

ExaWind

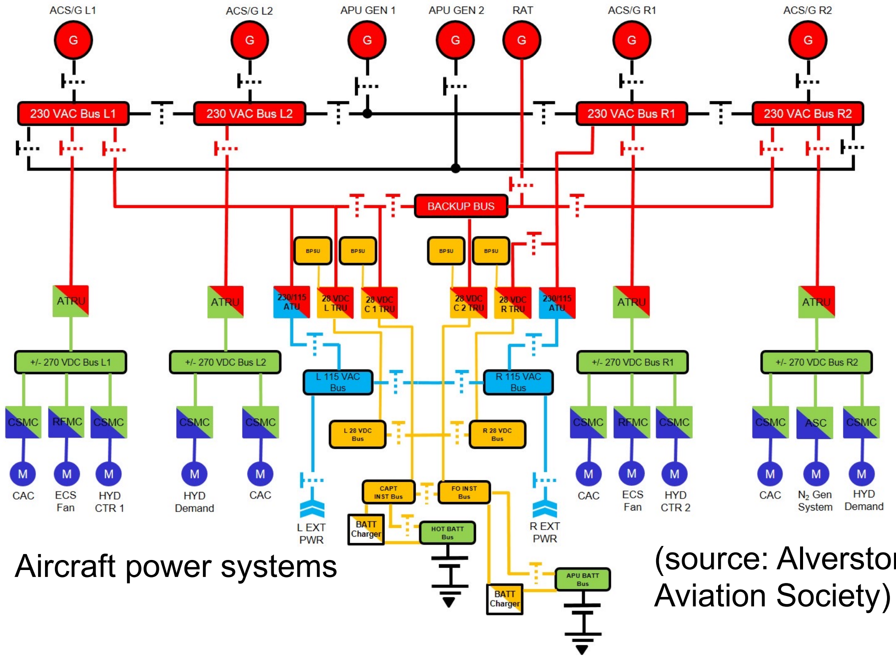
hypre solve performance on AMD GPUs 30-40% faster than Summit



Slide courtesy of Andrew Siegel and Erik Draeger

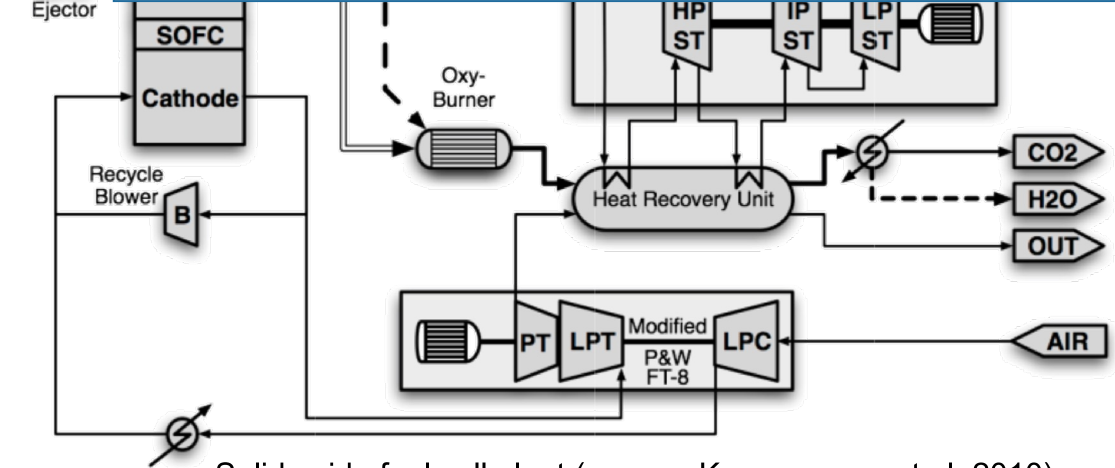
Systems Engineering Domain

- ExaSGD addresses systems engineering problems
- Produced new direct sparse solvers using non-supernodal structures, for GPUs
- Cholesky for symmetric, LU for non-symmetric
- Joint effort between SuperLU, Ginkgo, ExaSGD teams



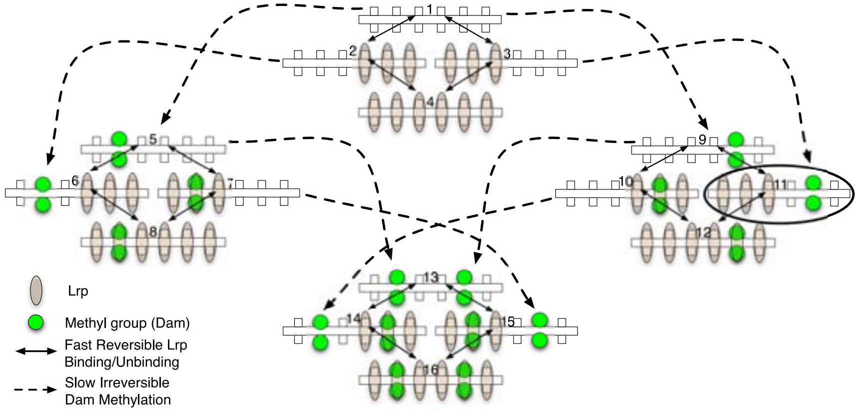
Aircraft power systems

(source: Alverstone Aviation Society)

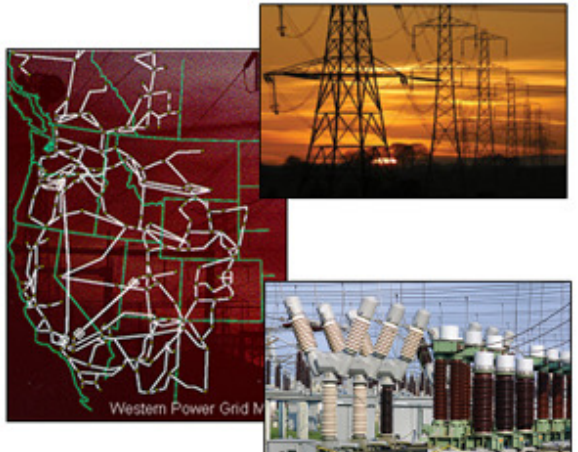


Solid oxide fuel cell plant (source: Kameswaran at al. 2010)

Buildings (source: EEB Hub, B661 2014)



Gene regulatory networks (source: Peles et al. 2006)



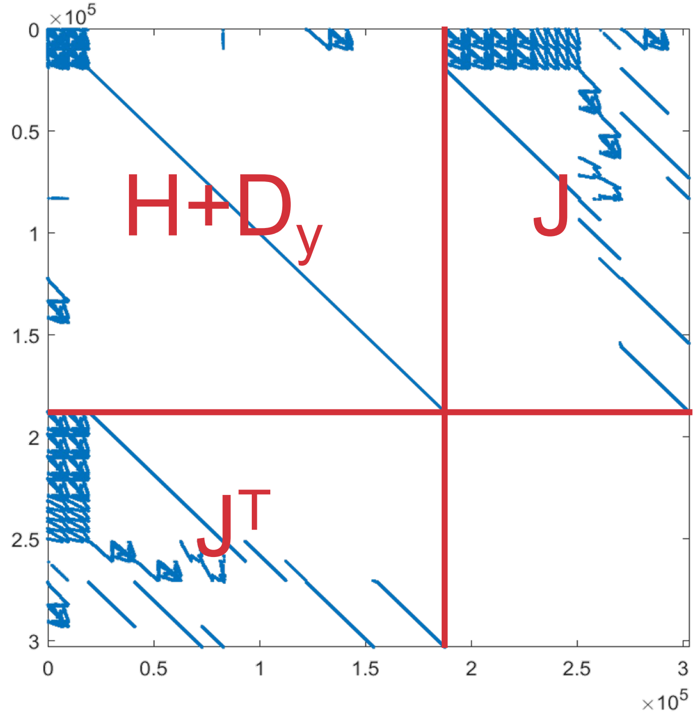
Power grids
(source: PNNL)

Underlying KKT Linear System Properties

- Security constrained optimal power flow analysis
- The interior method strategy leads to symmetric indefinite linear systems

$$\overbrace{\begin{bmatrix} H + D_y & J \\ J^T & 0 \end{bmatrix}}^{K_k} \overbrace{\begin{bmatrix} \Delta y \\ \Delta \lambda \end{bmatrix}}^{\Delta x_k} = \overbrace{\begin{bmatrix} r_y \\ r_\lambda \end{bmatrix}}^{r_k},$$

  - J – sparse constraints Jacobian,
  - H – sparse Hessian,
  - D_y – arises from log-barrier function

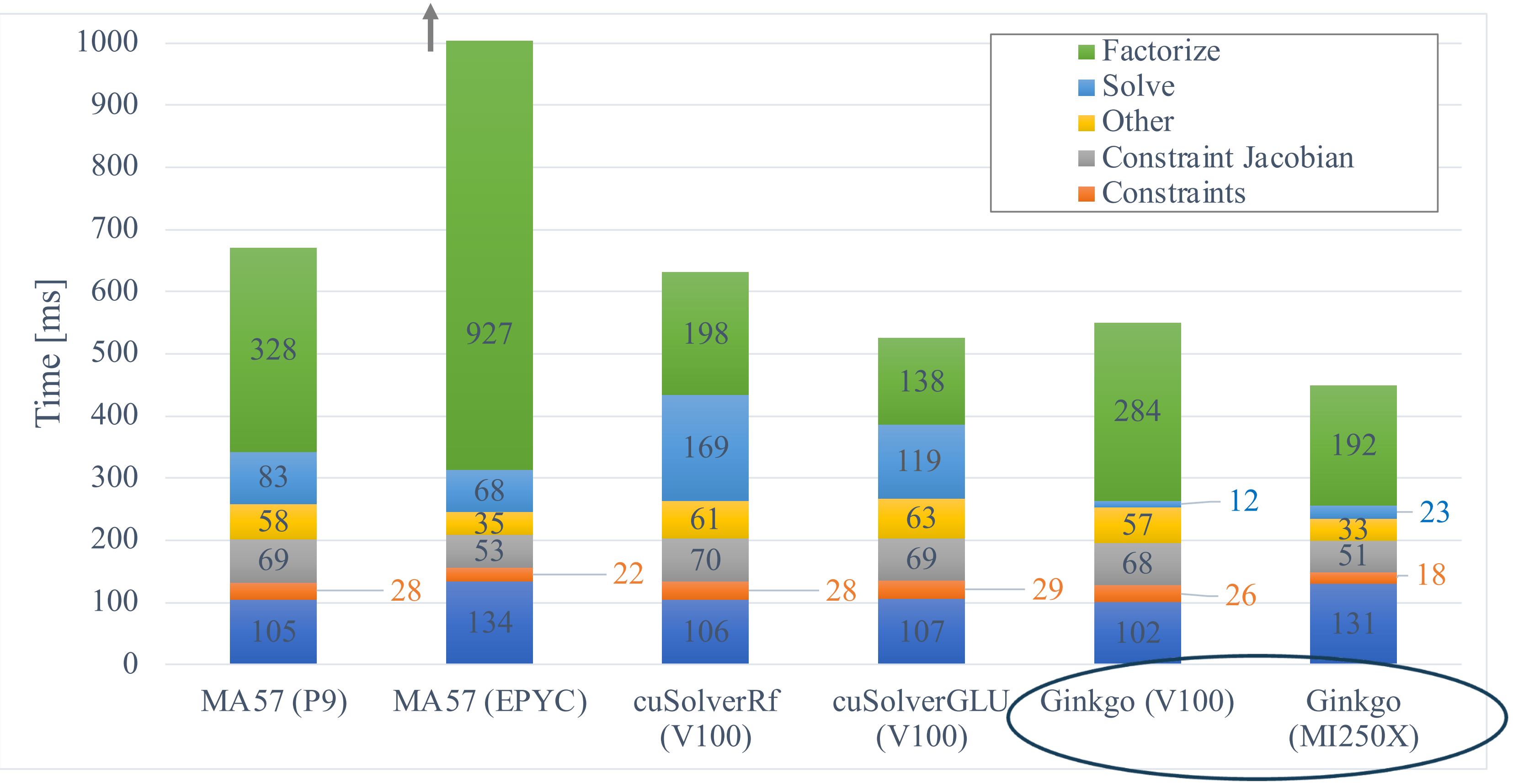


Typical sparsity pattern of optimal power flow matrices: No obvious structure that can be used by linear solver.

- The challenge: we need to solve a long sequences of such systems

Linear Solver Performance within Optimization Algorithm
Average per iteration times (including first iteration on CPU)

- Each GPU solution outperforms all CPU baselines
- Ginkgo performance improves on a better GPU
- Iterative refinement configuration affects linear solver performance and optimization solver convergence
- ***Ginkgo provides the first portable GPU-resident sparse direct linear solver for non-supernodal systems***



EXASCALE COMPUTING PROJECT

SZ Example of Success Story: Cosmology

ECP HACC: N-body problem with domain decomposition, medium/long-range force solver (particle-mesh method), short-range force solver (particle-particle/ particle-mesh algorithm).

Very hard to compress datasets (very little correlation, no smoothness)

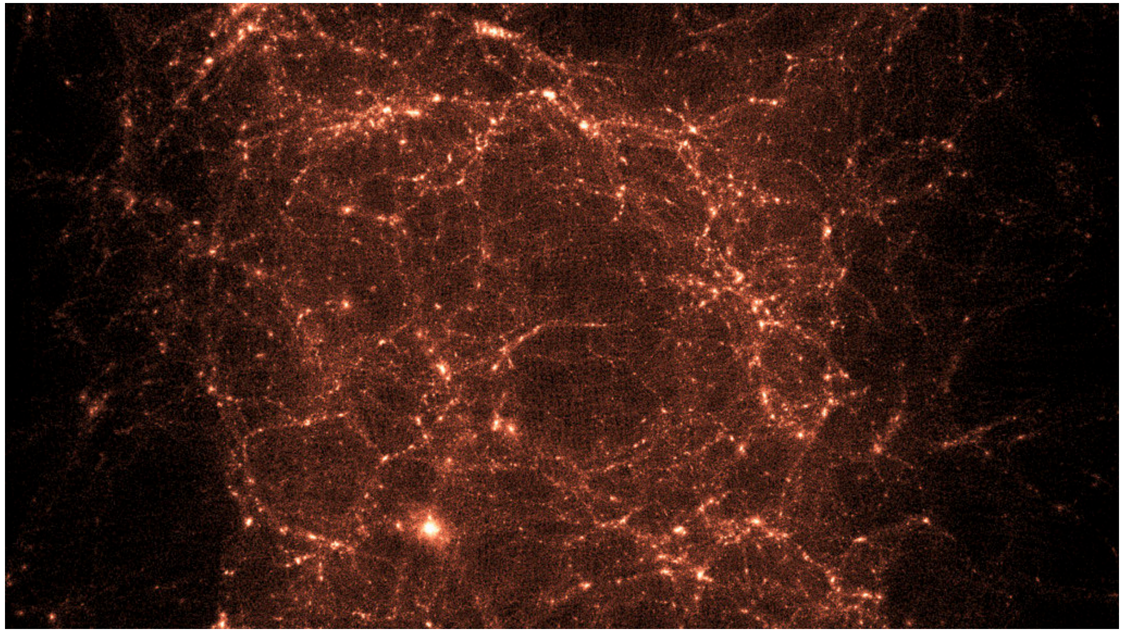
Particle dataset: 6 x 1D array (x, y, z, vx, vy ,vz)

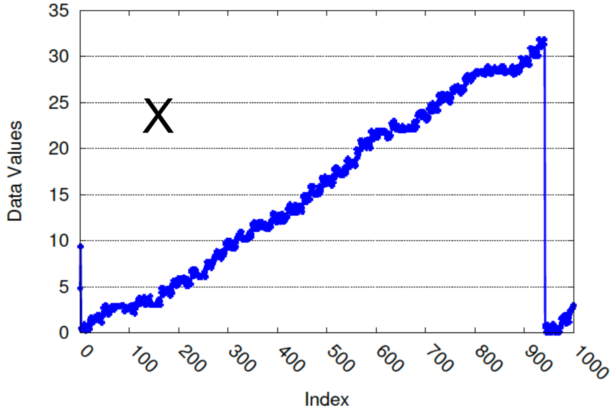
Preferred error controls:

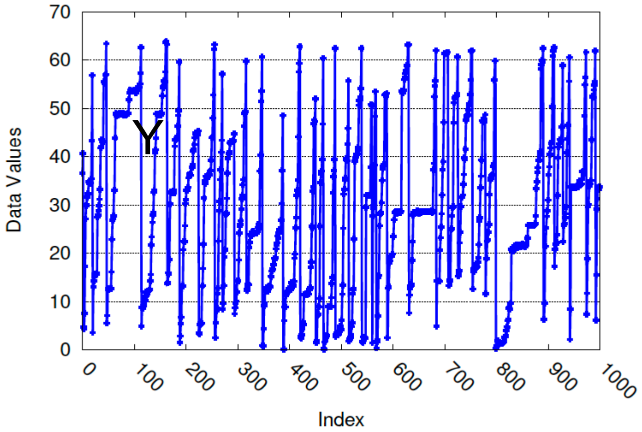
- Point wise max error (Relative) bound
- Absolute (position), Relative (Velocity)

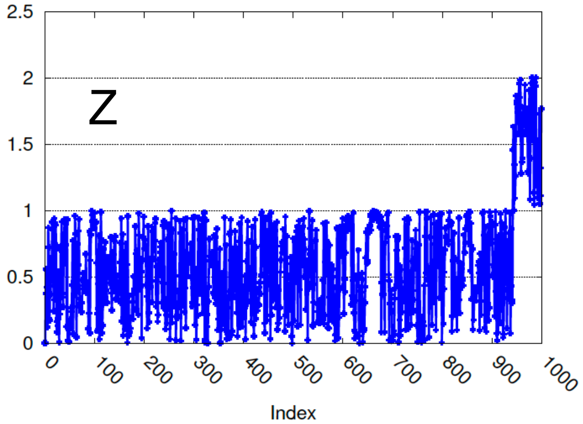
SZ: **CR ~5** (~6bits/value) at 10^{-3} error bound

ANL: Cosmological Simulations for Large-Scale Sky Surveys







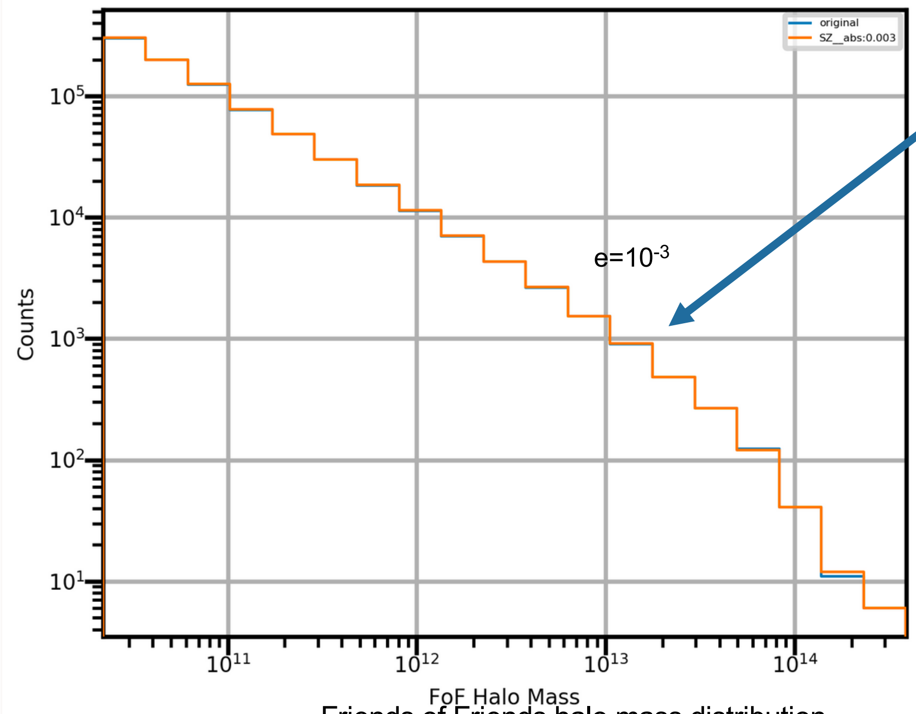


Figures from Cbench (ECP EXASKY)

SZ Example of Success Story: Cosmology

ECP HACC: Results validation from user's analysis

3kpc absolute error bound
(particle position)



Friends of Friends halo mass distribution

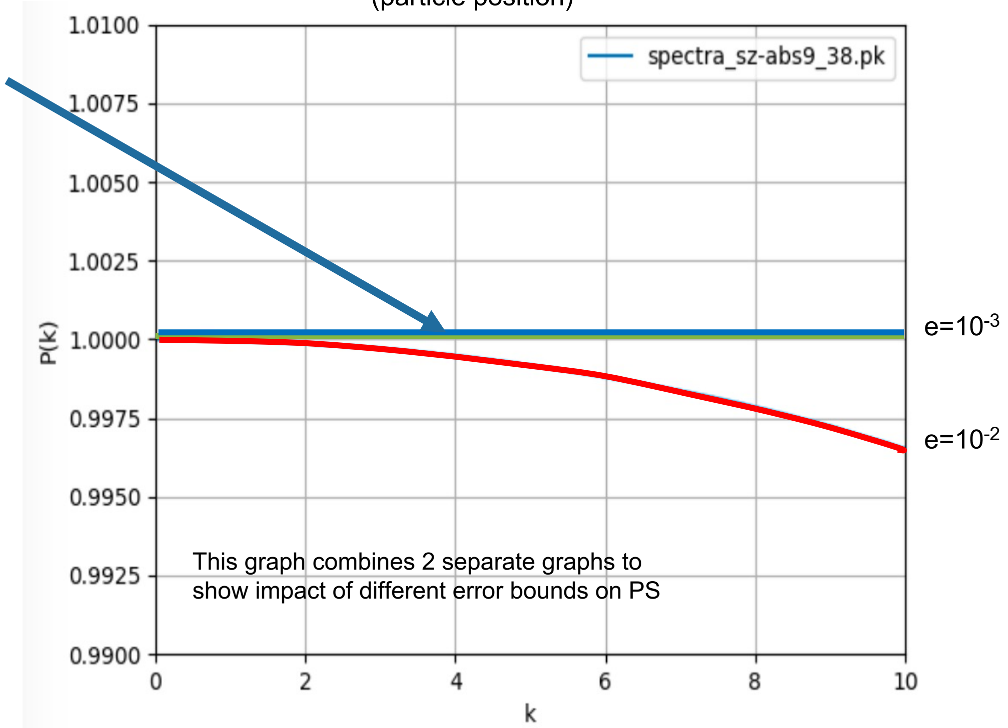
An error bound of 10^{-3} produces a near perfect overlap between the analysis on the initial and SZ compressed data

Original:

SZ (e=10^{-3}):

SZ (e=10^{-2}):

3kpc absolute error bound
(particle position)



Power Spectrum

→Reduced storage footprint by **~5x**

→**Allows saving 5x more snapshots**

What is Kokkos?

- Ecosystem for portable and performant parallel programming (focus: on-node parallelism)
  - **Expanding solution** for common needs of modern science/engineering codes
- **Kokkos Core: C++ Programming Model for Performance Portability**
  - Goal: Write algorithms once, run everywhere (almost) optimally
  - Implemented as a template library on top of CUDA, OpenMP, HIP, SYCL, …
  - Aligns with developments in the C++ standard
- **Kokkos Kernels: Numerical libraries with interfaces to vendor kernels**
  - Goal: Deliver high performance kernels that are portable
  - Optimized implementations for sparse/dense linear algebra and graph kernels
- **Open-Source Software**: https://github.com/kokkos
- Production use on all major HPC systems by dozens of teams: *Frontier, Summit, Fugaku*, ….
- Many users at a wide range of institutions:

ECP users of Kokkos

- Extensively used across ECP AD and ST projects
  - **Apps**: ExaWind, EXAALT, WDMApp, ExaAM, LatticeQCD, E3SM-MMF, SNL & LANL ATDM apps
  - **Codesign**: ExaGraph, CoPA
  - **ST**: ALExa (ArborX and DTK), Kokkos Kernels, Trilinos, FleCSI, PETSc

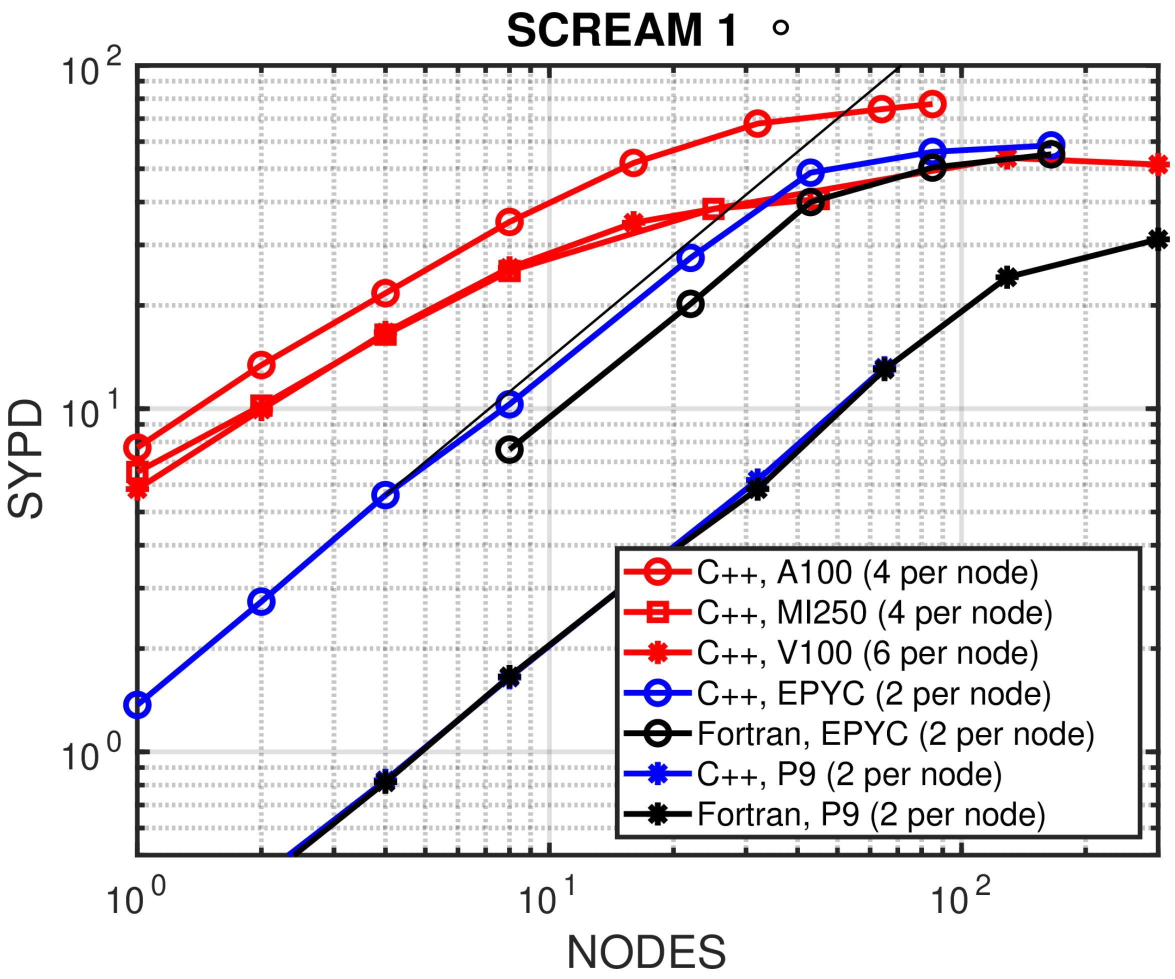
Kokkos and E3SM: Energy Exascale Earth System Model

Different components developed by different subteams

- Original code in Fortran, now C++
- Some use Kokkos some YAKL

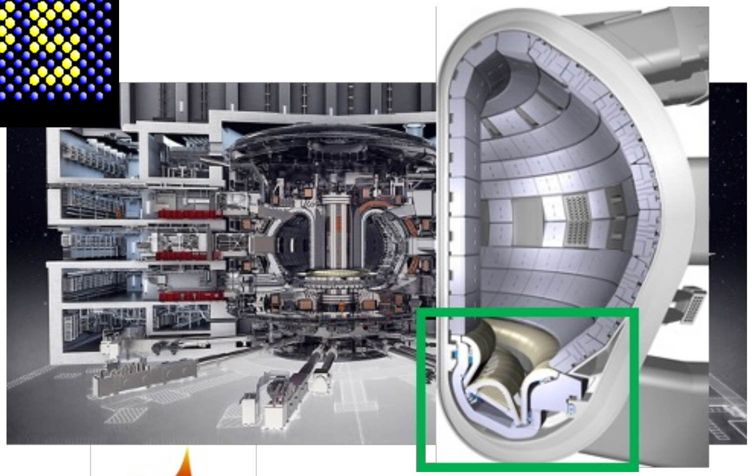
E3SM Atmosphere, running at 1 deg resolution: 128L, NH dycore, 10 tracers, P3/SHOC physics with prescribed aerosols, no convective parameterization

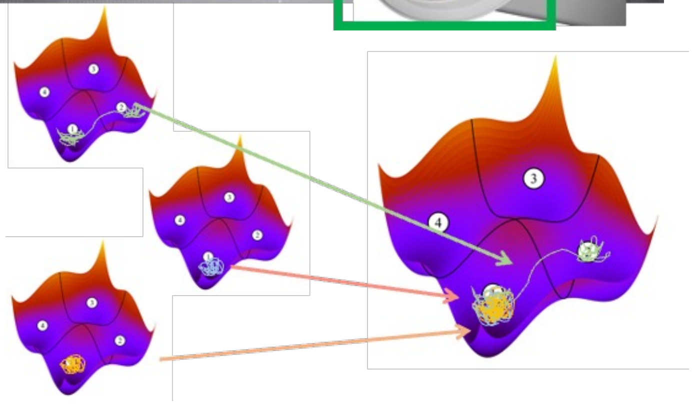
- Performance portability
  - IBM P9, AMD EYPC
  - NVIDIA V100, A100
  - AMD MI250
- CPU performance:
  - C++/Kokkos as fast or faster than Fortran
- GPU performance:
  - Large scaling range where GPU nodes are 4-10x faster than CPU nodes



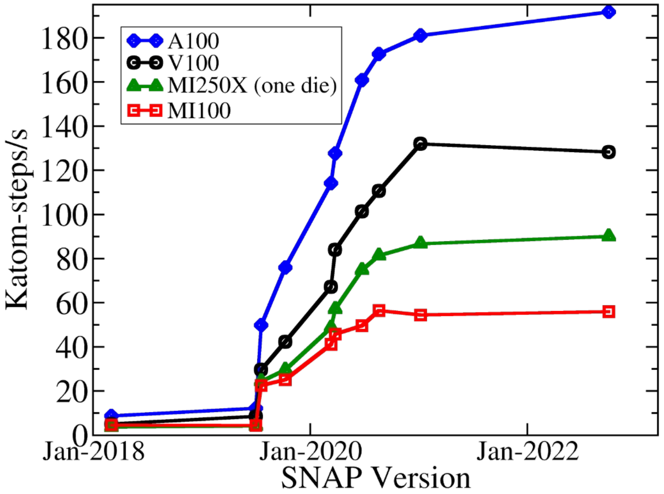
EXASCALE COMPUTING PROJECT

Kokkos and EXAALT/LAMMPS





- ECP EXAALT project seeks to extend **accuracy, length, and time** scales of simulations of plasma-material interactions for fusion energy
- EXAALT takes many replicas of LAMMPS and stitches them together to enable long-timescale simulations. **LAMMPS uses Kokkos for performance portability**
- Optimizing SNAP machine learning potential in LAMMPS for NVIDIA V100 gave ~25x speedup on AMD MI250X as well
- EXAALT team successfully executed their KPP1 simulation on 7000 Frontier nodes (~75% of full Frontier).
- Measured performance = **398.5x speedup over the Mira baseline**
- Extrapolation to full Frontier → **~530x** over Mira (ECP target was 50x)



EXASCALE COMPUTING PROJECT

Kokkos Participation in ISO C++ Standards Committee

MDSPAN

- mdspan got LEWG (working group in charge of library design review) approval for C++23
  - Still needs wording review approval
- mdspan will replace guts of Kokkos View for managing data layouts
- Strong support from vendors
- C++ will finally have multi dimensional arrays like Fortran!
  - But much better: incorporates all the customization points from Kokkos
  - Layouts, Memory Access Traits, Can also be used for memory space typesafety
- Proposal: https://wg21.link/P0009 Implementation: https://github.com/kokkos/mdspan

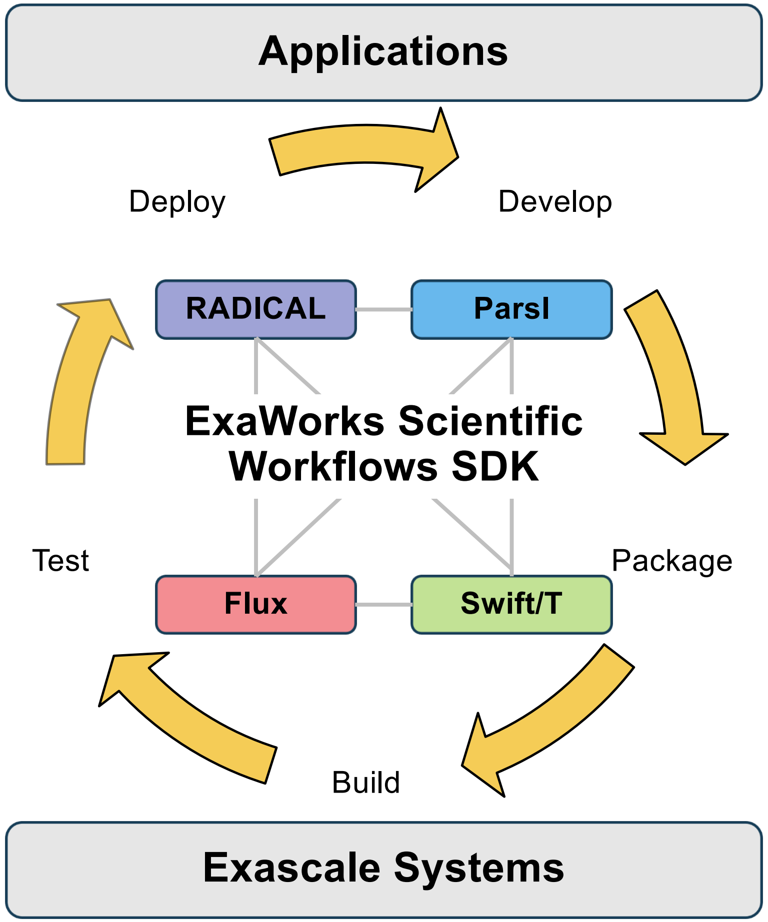
std::linalg

- Full BLAS for C++ but with mixed precision and flexible data layouts via mdspan
- Missed the deadline for C++23 due to limited committee review time, very likely for C++26
- Vendors are co-authors – collaborating with some on reference implementation

EXASCALE COMPUTING PROJECT

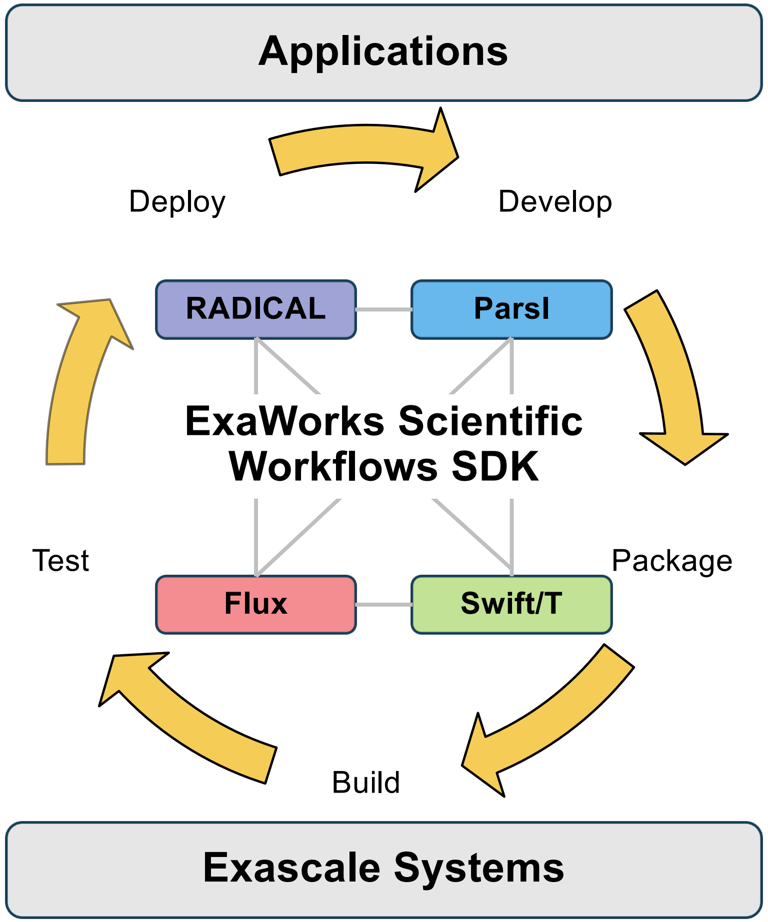
ExaWorks: Seed Technologies with ECP Application Impact

- Scientific workflows SDK includes four seed technologies
  - **Flux** – hierarchical resource and job management software
  - **Parsl** – flexible and scalable parallel programming library for Python
  - **RADICAL** – component-based workflow middleware
  - **Swift/T** – high performance dataflow computing
- Several existing collaborations with ECP applications
  - **CANDLE** – steering ensembles of molecular dynamics simulations
  - **ExaLearn** – multi-scale reinforcement learning workflow for materials
  - **ExaAM** – improve the scalability of the ExaConstit workflow
- **Already demonstrated ECP application impacts**
  - Winner and two of three finalists for Covid-19 Gordon Bell used ExaWorks
  - ExaAM reported **4x performance improvement** using Flux
- **Engaging the broader community on improving the SDK**
  - Advisory Board drawn from key stakeholders
  - Surveyed applications teams and are conducting deep-dive interviews on requirements
  - Working with the Facilities (ANL, ORNL, and NERSC) to **identify and address** their concerns



ExaWorks: Interoperable Components Expand Capabilities

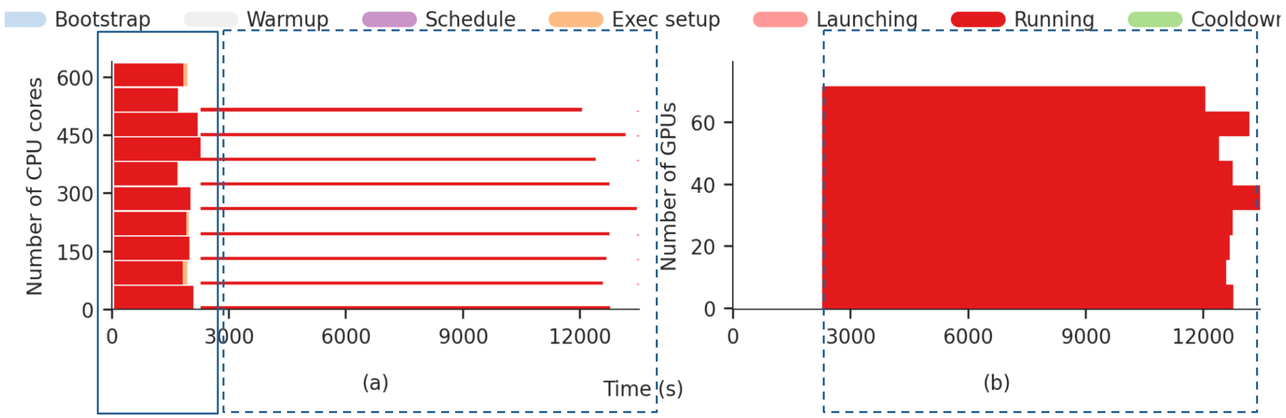
- Level 0: Technologies can be packaged together
- Level 1: Break down vertical silos and leverage capabilities
  - **Internal** – RADICAL and Parsl can use Flux, for example
  - **External** – prototype integration of Balsam components in Parsl
- Level 2: Community developed and supported component APIs
  - **J/Psi**: Job – Portable Submission Interface
    - Instantiated with **community design process**
    - Agreement from developers to adopt (e.g., Balsam)
- **Engaging the broader community on interoperability**
  - Workflows Community Summit in January 2021
  - Developing community policies for SDK inclusion (Level 0)
  - Identifying and implementing impactful integrations (Level 1)
  - Working together to develop meaningful APIs with reference implementations and adoption agreements (Level 2)



Results | Frontier

https://code.ornl.gov/matitov/radical-entk-frontier/-/tree/main/

- Workflow for OpenFOAM and ExaCA (ExaAM challenge problem) using ExaWorks
  - Standalone RADICAL-EnTK application (tested on Frontier, 10 nodes)



*Resource utilization on **Frontier** for a workflow comprised of OpenFOAM and ExaCA tasks.*

*- **Stage 1 (solid-line box):** Consists of 10 OpenFOAM MPI tasks; each task is an additiveFoam executable running on 60 CPU cores;*
*- **Stage 2 (dashed-line box):** Consists of 9 ExaCA MPI tasks; each task is an ExaCA-Kokkos executable running on 8 CPU+GPU.*

EXASCALE COMPUTING PROJECT

2.3.5.10 ExaWorks: Future Impacts

- Community curated, portable, scalable, interoperable, sustainable, and trusted scientific workflow ecosystem for **high-performance, exascale workflows**
- Demonstrate value of interoperability to stakeholders
  - **Improved performance** and capabilities
  - Leverage components to support new machines
- Realize benefits from inclusion in E4S
  - Use Spack recipes, build pipelines and caches, and testing infrastructure
  - **Improve testing** and confidence in technologies
  - **Simplify deployment** at Facilities and in user space – ‘spack install exaworks’
- Improve user experience with the technologies
- **Deliver application impacts** inside and outside of ECP
- **Foster a sustainable workflows community**

Responding to complexity: Software Ecosystem via Platforms

Takeaways from product sampler

- **Wide range of products and teams:** libs, tools, small personality-driven, large community-driven
- **Varied user base and maturity:** widely used, new, emerging
- **Variety of destinations:** direct-to-user, facilities, community stacks, vendors, facilities, combo of these
- **Wide range of dev practices and workflows:** informal to formal
- **Wide range of tools:** GitHub, GitLab, Doxygen, Readthedocs, CMake, autotools, etc.
- Question at this point might (should?) be:
  - *Why are you trying to make a portfolio from this eclectic assortment of products?*
- Answer:
  - Each product team charged with challenging tasks:
    - Provide capabilities for next-generation leadership platforms
    - Address increasing software quality expectations
    - While independently developed, product compatibility and complementarity improvements matter
  - **Working together on these frontiers is better than going alone**

Software Platforms: “Working in Public” Nadia Eghbal

- Platforms in the software world are digital environments that intend to improve the value, reduce the cost, and accelerate the progress of the people and teams who use them
- Platforms can provide tools, workflows, frameworks, and cultures that provide a (net) gain for those who engage
- Eghbal Platforms:

	HIGH USER GROWTH	LOW USER GROWTH
HIGH CONTRIBUTOR GROWTH	Federations (e.g., Rust)	Clubs (e.g., Astropy)
LOW CONTRIBUTOR GROWTH	Stadiums (e.g., Babel)	Toys (e.g., ssh-chat)

Eghbal, Nadia. Working in Public: The Making and Maintenance of Open Source Software (p. 60). Stripe Press. Kindle Edition.

About Platforms and ECP

- The ECP is commissioned to provide new scientific software capabilities on the frontier of algorithms, software and hardware
- The ECP provides platforms to foster collaboration and cooperation as we head into the frontier:
  - **E4S:** a comprehensive portfolio of ECP-sponsored products and dependencies
  - **SDKs:** Domain-specific collaborative and aggregate product development of similar capabilities

Delivering an open, hierarchical software ecosystem

More than a collection of individual products

Extreme-scale Scientific Software Stack (E4S)

- E4S: HPC software ecosystem – a curated software portfolio
- A **Spack-based** distribution of software tested for interoperability and portability to multiple architectures
- Available from **source**, **containers**, **cloud, binary caches**
- Leverages and enhances SDK interoperability thrust
- Not a commercial product – an open resource for all
- Growing functionality: May 2023: E4S 23.05 – 100+ full release products

Community Policies
Commitment to SW quality

DocPortal
Single portal to all E4S product info

Portfolio testing
Especially leadership platforms

Curated collection
The end of dependency hell

Quarterly releases
Release 23.05 – February

Build caches
10X build time improvement

Turnkey stack
A new user experience

https://e4s.io

Post-ECP Strategy
LSSw.io, PESO

https://spack.io

Spack lead: Todd Gamblin (LLNL)

https://e4s.io

E4S lead: Sameer Shende (U Oregon)

Also includes other products, e.g.,
AI: PyTorch, TensorFlow, Horovod
Co-Design: AMReX, Cabana, MFEM

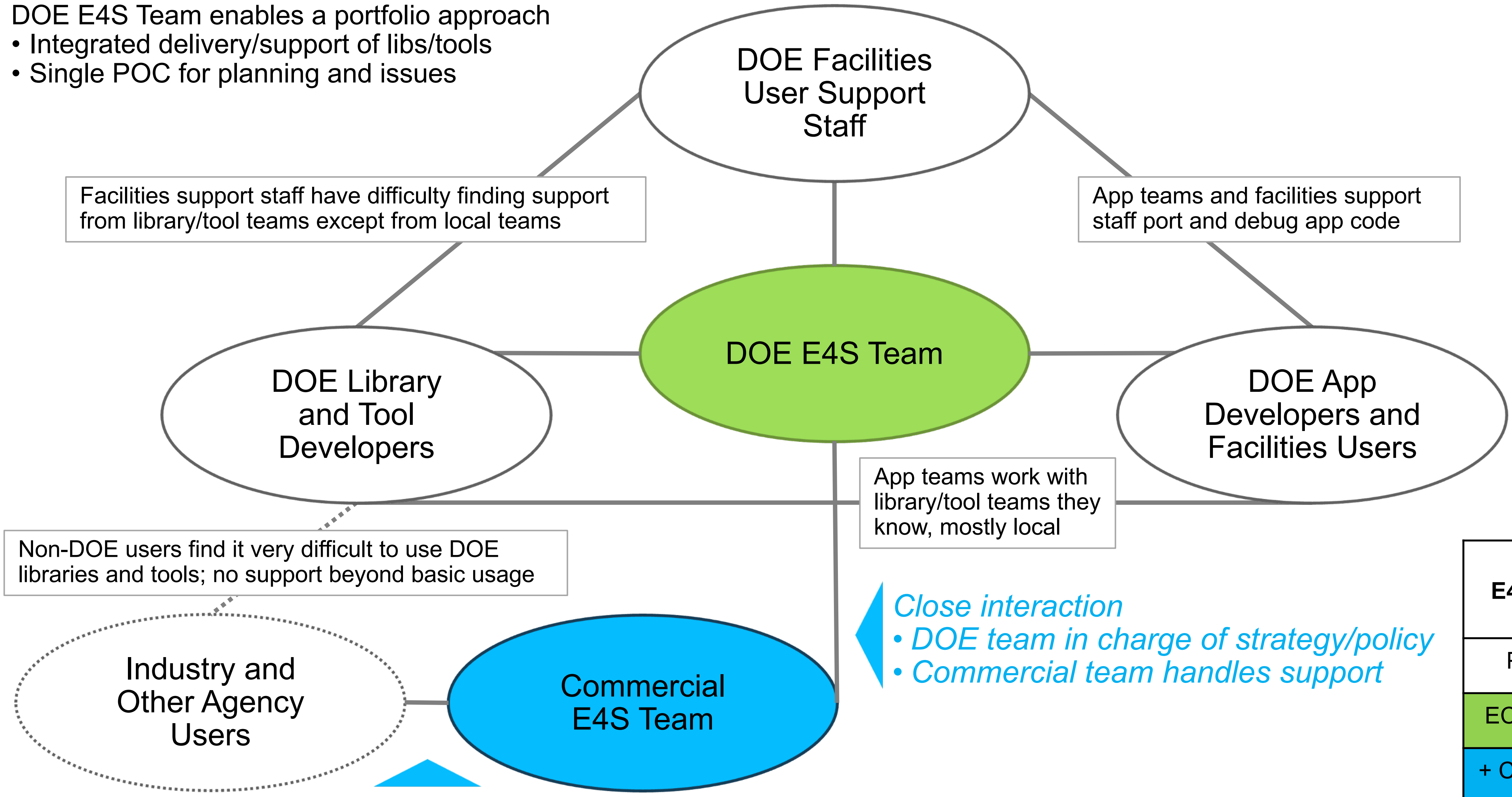
EXASCALE COMPUTING PROJECT

Spack

- E4S uses the Spack package manager for software delivery
- Spack provides the ability to specify versions of software packages that are and are not interoperable
- Spack is a build layer for not only E4S software, but also a large collection of software tools and libraries outside of ECP ST
- Spack supports achieving and maintaining interoperability between ST software packages
- https://spack.io

E4S Business Model: Optimize Cost & Benefit Sharing

DOE E4S Team enables a portfolio approach
- Integrated delivery/support of libs/tools
- Single POC for planning and issues



First of a kind interactions
- *Industry/agencies can acquire support*
- *Shared costs and benefits with DOE*

E4S Phase	Cost & Benefit Scope
Pre-E4S	Local Facility
ECP support	DOE complex
+ Commercial support	Universal

Commercial E4S Support Essential for non-DOE Users

Provides vehicle for sustainable non-DOE user support

Support Phase	Primary Scope	Primary Cost and Benefit Sharing Opportunities
Pre-E4S	Local facility	**Local costs and benefits:** Prior to ECP and E4S, libraries and tools were typically strongly connected to the local facility: ANL libs and tools at ALCF, LBL at NERSC, LLNL at Livermore Computing, etc.
+ ECP E4S	All DOE facilities	**DOE complex-shared costs and benefits:** ECP requires, and E4S enables, interfacility availability and use of libs across all facilities: First-class support of ANL libs and tools at other facilities, etc.
+ Commercial E4S	DOE facilities, other US agencies, industry, and more	**Universal shared costs and benefits:** Commercial support of E4S expands cost and benefit sharing to non-DOE entities: DOE costs are lower, software hardening more rapid. US agencies, industry and others can contract for support, gaining sustainable use of E4S software and contributing to its overall support.

E4S 23.05: What's New?

- E4S includes support for Intel oneAPI 2023.1 software (BaseKit and HPCToolkit) in containers on x86_64 with support for HPC packages built with Intel compilers
- E4S includes support for CUDA architectures
  - 70 (V100), 80 (A100), and 90 (H100) under x86_64
  - 70 under ppc64, and
  - 75 and 80 under aarch64
- E4S includes supports ROCm for gfx908 (MI100) and gfx90a (MI200) architectures under x86_64
- E4S includes support for DOE LLVM under x86_64, ppc64le, and aarch64
- E4S includes new applications: Xyce (with pymi), LBANN, Quantum Espresso, LAMMPS, WARPX, Dealii, and OpenFOAM.
- E4S includes support for AI/ML frameworks such as TensorFlow and PyTorch support for A100 as well as H100 GPUs is integrated in E4S 23.02
- E4S supports updates to 100+ HPC packages on x86_64, aarch64, and ppc64le, 100K+ binaries in E4S Spack Build Cache
- New E4S tools: e4s-alc (à la carte) customizes container images, e4s-cl (container launch) replaces MPI at runtime!
- Detailed documentation for installing E4S on bare-metal and using containers

E4S Engagements: DOE, Other US Agencies

- DOE
  - NERSC, OLCF, ALCF – Active porting on leadership, exascale platforms
  - Multiple ECP apps: ExaWind, WDPApp, Cinema
  - Emerging Sandia effort: Xyce on E4S on AWS for a summer class
- NSF
  - E4S installed on Frontera, TACC; Bridges-2, PSC; BlueWaters, NCSA; Expanse, SDSC
  - SDSC: E4S Singularity containers available on Open Science Grid High Throughput Computing (https://OSG-HTC.org)
- NOAA
  - E4S base images being used in production on AWS and in custom containers
- DoD
  - Testing installation of E4S on Narwhal, Navy DSRC
- NASA
  - Singularity support for E4S on Pleiades
  - Custom E4S images exploration
  - Day-long workshop, July 18

E4S Engagements: International

- CEA, France: E4S engagement discussed with CEA
  - Workshop planned in July 2023 with ParaTools, SAS
- CSC, Finland: Lumi Supercomputer
  - E4S Workshop in March 2023
  - https://ssl.eventilla.com/event/WL761
  - E4S 23.02 installed on Lumi
- Pawsey Supercomputing Center, Perth, Western Australia
  - E4S workshop planned in April 2023
  - https://pawsey.org.au/event/evaluate-application-performance-using-tau-and-e4s-april-4-5/
  - E4S 23.02 installed on Setonix

- E4S provides a large stack of reusable software libraries and tools
  - Build from scratch using Spack, or use via containers, cloud, build caches
  - Makes stack management easier, portable, lower cost
- We expect E4S to be one of the most important legacies of ECP

Steady Stream of E4S to ALL Facilities!

GitHub

ST Development Team

ST Development Team works any issues reported

PHASE I

Product establishes:

1. Spack-based package and build
2. Validation testing in E4S testsuite
3. Documentation for install and use
4. Accessible public repository

E4S community policies

E4S

PHASE II

- E4S establishes install at facilities
- E4S packages get tested and validated in facility environment
- New E4S releases automatically tested through ECP CI infrastructure

OUTPUT

- **High-quality Spack recipes, for ECP products, ready for facility systems**

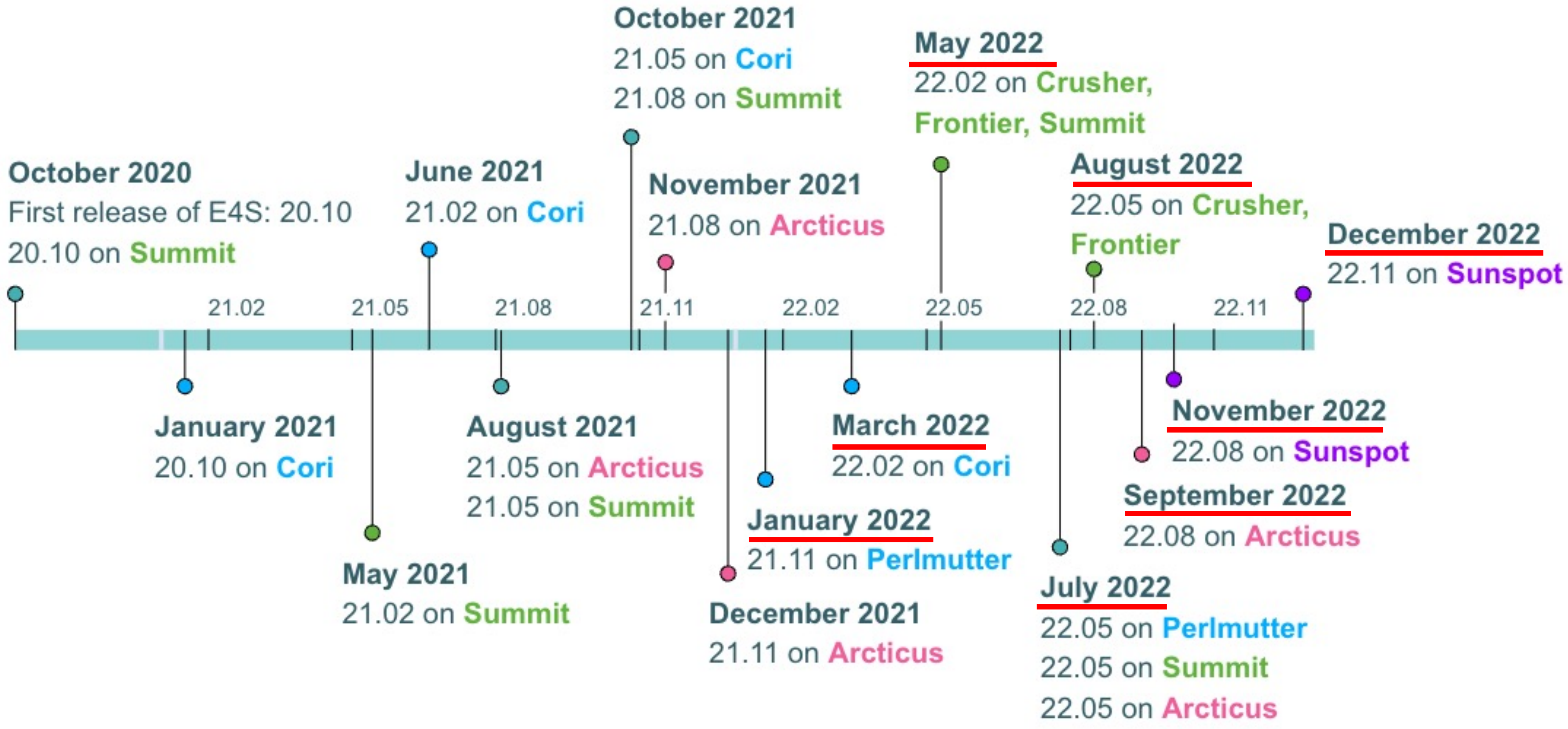
E4S

FEEDBACK PHASE

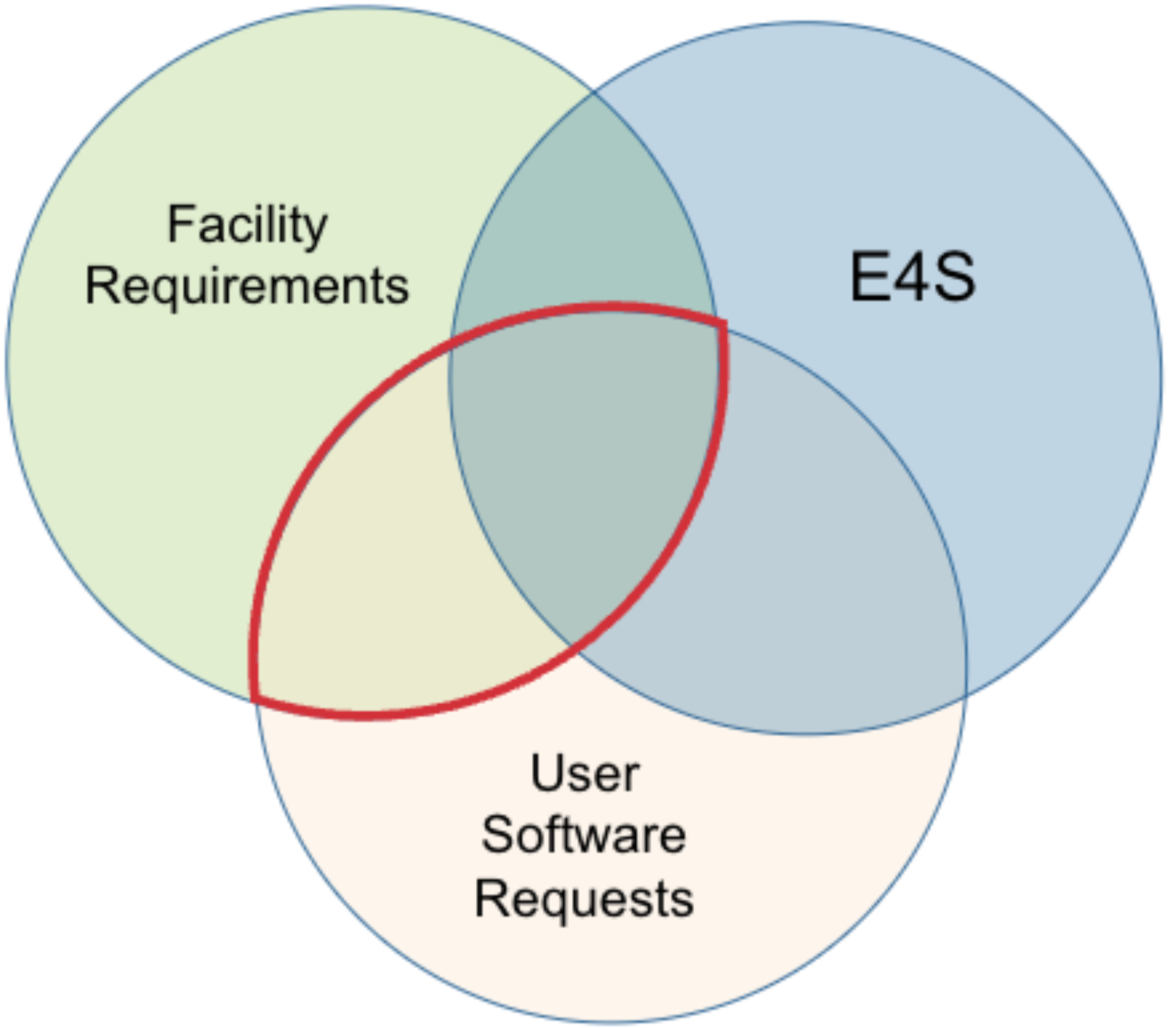
- **Software Integration team integrates packages into facility system**
- **New E4S release up-streamed and support requests from facility generated as needed**
- **Issues/Fixes/changes worked with developers as needed**

EXASCALE COMPUTING PROJECT

E4S Release and Integration Timeline



Facility Software Integration



- Not all E4S products can be maintained by facility staff (there are a lot!)
- User requests drive facility priorities
- Compatibility and maintainability, with facility environments, are essential
- Red area depicts area of focus from perspective of software integration staff at facilities
- ‘Level 2’ Support from ParaTools helps!

E4S Community Policies: *A commitment to quality improvement*

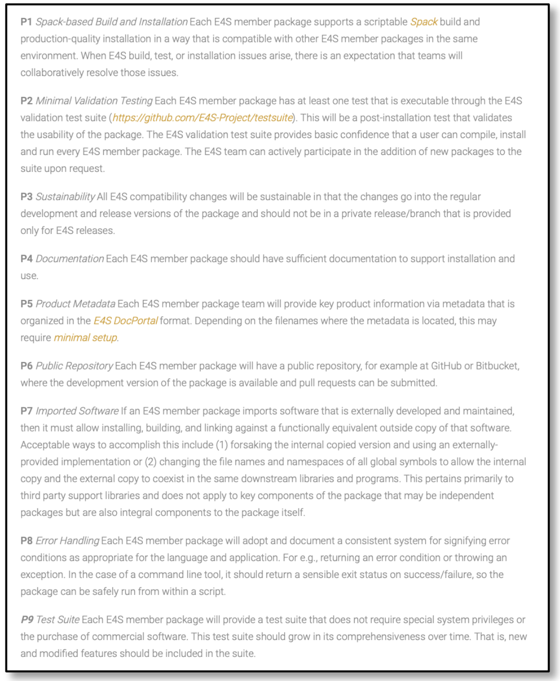
- Enhance sustainability and interoperability
- Serve as membership criteria for E4S
  - Membership is not required for *inclusion* in E4S
  - Also includes forward-looking draft policies
- Modeled after xSDK community policies
- Multi-year effort led by SDK team
  - Included representation from across ST
  - Multiple rounds of feedback incorporated from ST leadership and membership

SDK lead: Jim Willenbring (SNL)

Policies: Version 1

https://e4s-project.github.io/policies.html

- **P1: *Spack-based Build and Installation***
- **P2: *Minimal Validation Testing***
- **P3: *Sustainability***
- **P4: *Documentation***
- **P5: *Product Metadata***
- **P6: *Public Repository***
- **P7: *Imported Software***
- **P8: *Error Handling***
- **P9: *Test Suite***



P1 *Spack-based Build and Installation* Each E4S member package supports a scriptable *Spack* build and production-quality installation in a way that is compatible with other E4S member packages in the same environment. When E4S build, test, or installation issues arise, there is an expectation that teams will collaboratively resolve those issues.

P2 *Minimal Validation Testing* Each E4S member package has at least one test that is executable through the E4S validation test suite (*https://github.com/E4S-Project/testsuite*). This will be a post-installation test that validates the usability of the package. The E4S validation test suite provides basic confidence that a user can compile, install and run every E4S member package. The E4S team can actively participate in the addition of new packages to the suite upon request.

P3 *Sustainability* All E4S compatibility changes will be sustainable in that the changes go into the regular development and release versions of the package and should not be in a private release/branch that is provided only for E4S releases.

P4 *Documentation* Each E4S member package should have sufficient documentation to support installation and use.

P5 *Product Metadata* Each E4S member package team will provide key product information via metadata that is organized in the *E4S DocPortal* format. Depending on the filenames where the metadata is located, this may require *minimal setup*.

P6 *Public Repository* Each E4S member package will have a public repository, for example at GitHub or Bitbucket, where the development version of the package is available and pull requests can be submitted.

P7 *Imported Software* If an E4S member package imports software that is externally developed and maintained, then it must allow installing, building, and linking against a functionally equivalent outside copy of that software. Acceptable ways to accomplish this include (1) forsaking the internal copied version and using an externally-provided implementation or (2) changing the file names and namespaces of all global symbols to allow the internal copy and the external copy to coexist in the same downstream libraries and programs. This pertains primarily to third party support libraries and does not apply to key components of the package that may be independent packages but are also integral components to the package itself.

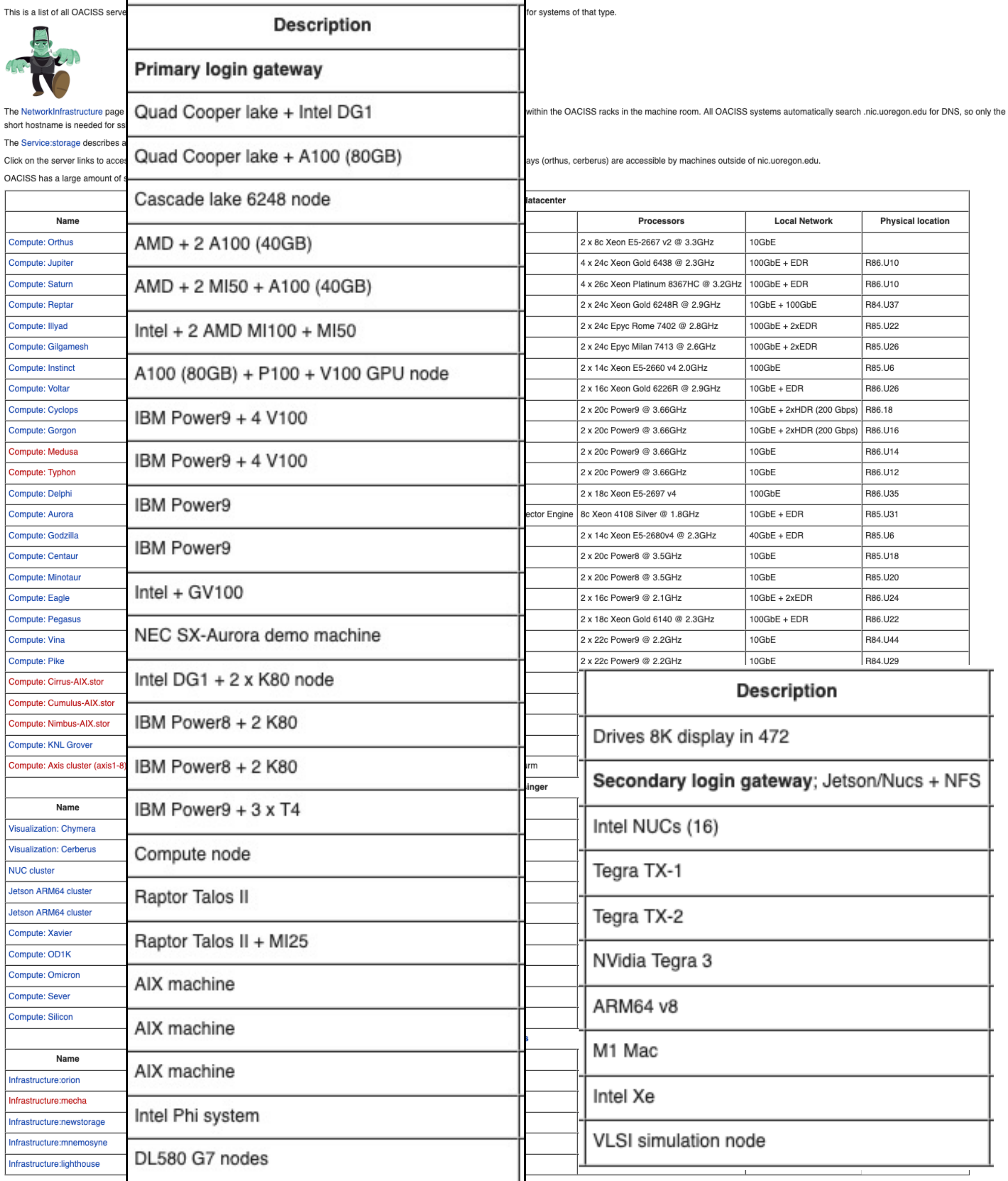
P8 *Error Handling* Each E4S member package will adopt and document a consistent system for signifying error conditions as appropriate for the language and application. For e.g., returning an error condition or throwing an exception. In the case of a command line tool, it should return a sensible exit status on success/failure, so the package can be safely run from within a script.

P9 *Test Suite* Each E4S member package will provide a test suite that does not require special system privileges or the purchase of commercial software. This test suite should grow in its comprehensiveness over time. That is, new and modified features should be included in the suite.

xSDK

Frank – Designed for Libs & Tools Developers

- Prep system for ECP libs & tools
- Access to latest non-NDA HW/SW
- Shared file system – 1 copy of SW
- Port to many device types at once
- Porting support from E4S team
- CI testing workhorse (500K builds)
- Next: Bare metal, BIOS-changing support for low-level software work



GPU Efforts Summary

- **One legacy** of ECP & E4S will be a SW stack that is **portable across Nvidia, AMD, and Intel** GPUS
- Porting to modern GPUs **requires almost everything** to be done **on the GPUs**
- Common refactoring themes:
  - Async under collectives
  - Batch execution – esp linear algebra
  - Pre-allocation and highly concurrent assembly: Sparse matrix assembly via COO format with atomics
- Two portability models (and hybrid of both) are used:
  - **Use portability layers:** Kokkos, RAJA or (eventually) OpenMP w target offload (OpenACC?)
  - **Isolate & and custom write:** Isolate perf-portable kernels and write your own CUDA, HIP, SYCL backend
  - **Hybrid:** Use portability layers, customize key kernels only
- Explore low-precision arithmetic: Substantial benefit (and risks)
- Rely more on third-party reusable libraries and tools – ECP apps are doing this

Leveraging the Future Potential of ECP Investments

100X

Frontier

Compute Node
1 64-core AMD “Optimized 3rd Gen EPYC” CPU
4 AMD Instinct MI250X GPUs
CPU & GPUs fully connected AMD Infinity Fabric

Node Memory
512 GiB HBM2e memory
512 GiB DDR4 memory
Cache Coherent Memory across entire node

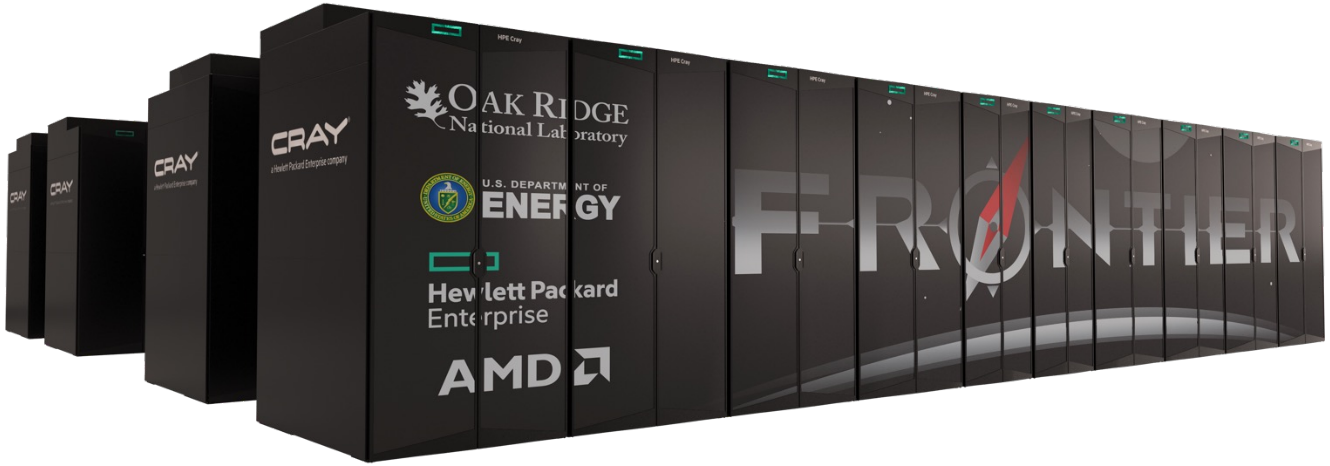
System Interconnect
HPE Slingshot. Four 200 Gbps (25 GB/s) NICs per node provides a node-injection bandwidth of 800 Gbps (100 GB/s)

High-Performance Storage
716 PB at 9.4 TB/s plus 11 Billion IOPS from
36 PB node local storage at 65 TB/s

Programming Models
MPI, OpenMP, OpenACC, HIP, C/C++, Fortran, DPC++, RAJA, Kokkos, and others

Node Performance
214 TF double precision

System Size
9,472 nodes



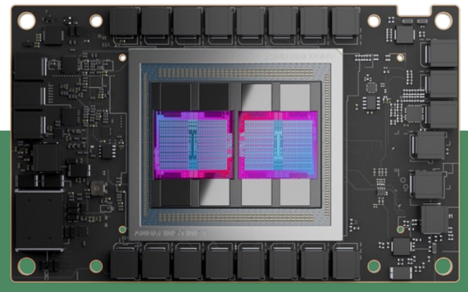
FRONTIER
HPE Cray EX

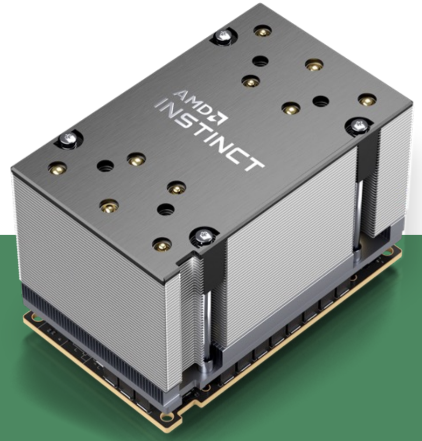
PEAK PERFORMANCE
≥2.0 Exaflop DP

FRONTIER COMPUTE NODE
1 64-core AMD CPU
4 AMD MI250X GPUs
4 TB NVM local storage

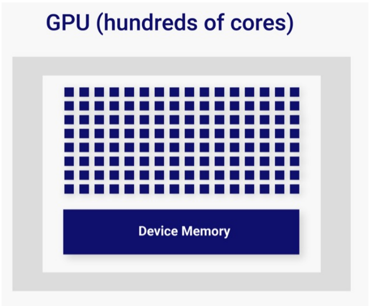
Source of all performance potential – BW and ops

Debuting in 2022 at 1.1 EF, Oak Ridge National Laboratory’s Frontier supercomputer is the world’s first exascale system. ORNL’s long history of supercomputing excellence enables scientists to expand the scale and scope of their research, solve complex problems in less time, and fill critical gaps in knowledge.



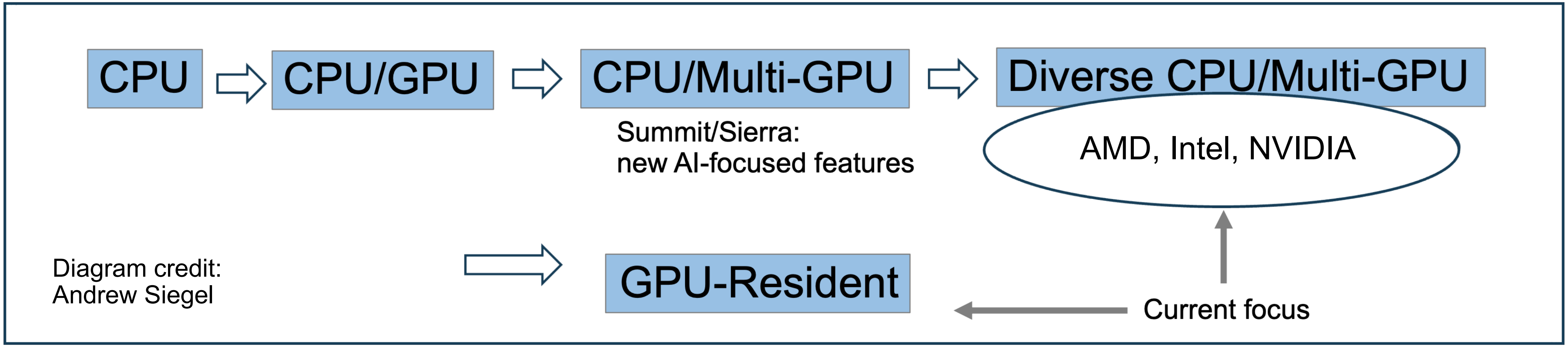


Heterogeneous accelerated-node computing



Accelerated node computing: Designing, implementing, delivering, & deploying agile software that effectively exploits heterogeneous node hardware

- Execute on the largest systems … AND on today and tomorrow's laptops, desktops, clusters, …
- We view *accelerators* as any compute hardware specifically designed to accelerate certain mathematical operations (typically with floating point numbers) that are typical outcomes of popular and commonly used algorithms. We often use the term GPUs synonymously with accelerators.



100X* your impact: Leveraging DOE/ECP investments

- Many communities still largely using home-grown solutions with room to improve. Opportunities:
  - Migrate from CPU to GPU – For scale out to larger problems, or scale in, to smaller GPU-enable systems (e.g., laptop)
  - Introduce modern software tools, workflows – leverage the outreach, training and culture focused on improvement
  - Integrate into larger software communities – E4S, xSDK, other software product communities
- How can we engage with these communities to realize the 100X improvement in science impact?
- DOE/ECP provides libraries, tools, expertise, and community connections that can be leveraged to realize 100X
- What are the best opportunities?
- What are the impediments?
- Can we produce strategic and tactical plans?
- Selling libraries and tools directly is hard to do …
- Selling 100X impact across DOE, other agencies and industry could be much easier

*100X (verb): *to increase (your scientific impact) by two orders of magnitude*

Opportunities to realize 100X

- Port to full use of GPUs:
  - Hotspot use of GPUs is a start but not sufficient.
  - Scalability very limited and capped for future GPU devices
- Utilize Spack ecosystem:
  - Opens ready access to hundreds of curated libraries and tools
  - Makes your code easy to consume if you publish Spack recipes for your code
  - Utilize Spack build caches (10X speedup in rebuild times)
- Utilize E4S
  - Curated libraries, tools, documentation, build caches, and more
  - Commercial support via ParaTools
  - Pre-built containers, binaries,
  - Cloud instances for AWS, Google – Permit elastic expansion, neutral collaboration for cross-agency work
- Leverage ECP team experience:
  - Engage with ECP staff at NASA-DOE events (next one is July 18)

100X Recipe

- Ingredients
  - A compelling science impact story
  - $$ - $$$
  - Staff
  - Computing resources, training
  - The deliverables and experience from DOE/ECP
  - Delivered via post-ECP organizations like PESO
  - And more…
- Steps
  - Translate science story to strategy and plan – leverage experience from ECP, others
  - ID node-level parallelization strategy – CUDA, HIP, DPC++, Kokkos, RAJA, OpenMP, others
  - Survey existing libraries and tools – Vendors, E4S, others
  - Explore available platforms – DOE Facilities, cloud, others
  - Leverage existing software ecosystem – containers, Spack, others
  - Leverage software communities – Product communities, communities of practice, others
  - Construct new codes within the broader ecosystem
  - Produce new science results

More than one way to leverage 100X

- 100X can be realized as exciting new science capabilities at the high end
  - Fundamental new science on leadership platform
  - New opportunities on affordable machines that fit in current data centers
- But can also reduce costs by running same problems 100X cheaper
- Migration to accelerated platforms can be used to
  - Migrate a problem from an HPC cluster to a deskside or laptop systems
  - Lower your AWS monthly charges – E4S is available for container/cloud
  - Keep energy costs in check while still growing computing capabilities

The E4S Two-Step

- Step 1:
  - Migrate existing MPI-CPU code on top of E4S:
    - All E4S libraries & tools compile & run well on CPU architectures, including multi-threading & (improving) vectorization
    - Pick a performance portability approach (as described above)
    - Rewrite your loops for parallel portability, e.g., rewrite in Kokkos or RAJA
    - Link against E4S CPU versions of relevant libraries
  - Potential benefits:
    - Migrating to E4S on a stable computing platform, easy to migrate incrementally and detect execution diffs
    - Single build via Spack
    - Potential for using build caches (10x rebuild time improvement)
    - Single point of access to documentation (E4S DocPortal)
    - Increased quality of user experience via E4S support, E4S and SDK quality commitments
    - Preparation for Step 2…

The E4S Two-Step

- Step 2: Turn on GPU build
  - Builds with GPU backends (especially if using Kokkos or RAJA)
  - Transition to GPU is a debugging and adaptation exercise
  - Track growth in E4S GPU capabilities as E4S products improve GPU offerings
- Consider interactions with E4S commercial support team
  - Pay someone for support
  - Get advice on product choices
    - DOE teams generally can’t give you good advice on which solver or IO library to use
      - Like asking Microsoft and Apple to tell whether to purchase a PC or Mac

Summary

- Using a portfolio-based approach for HPC software is about **going together vs going alone**
- While products vary greatly, we all face the **same frontiers**: **Evolving demands and systems**
- Success on the frontier is important for all HPC configurations: **leadership to laptop**
- The new and evolving E4S and SDK platforms enable **better, faster and cheaper**, in net
- A collective approach, E4S, enables **new relationships with facilities, vendors, apps, industry**
- Discussions with other US agencies progressing: NASA, NOAA, NSF – my hope: A national stack
- Potential NASA interests from ECP libraries and tools efforts
  - Spack – can be transformative itself, independent of E4S
  - Latest MPI, IO capabilities – available in E4S first
  - Flang – IMO the future of Fortran hinges on the success of Flang, NASA engagement matters
  - Kokkos/RAJA portability layers – lessons learned, starting point for your own layer, direct use
  - Lessons learned in new algorithms for highly-concurrent nodes
  - Longer term: leverage E4S and SDKs for better/faster/cheaper use of open source

Next Opportunity: July 18, 2023 NASA-ECP Workshop

- NASA and ECP will host a day-long workshop for NASA scientists on leveraging ECP efforts
- Speakers will provide overviews and some deep dives on capabilities with time for Q&A
- If you are interested, please contact:
  - Suzy Tichenor tichenorsp@ornl.gov
  - David Martin dem@alcf.anl.gov

Thank you

https://www.exascaleproject.org

This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy's Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation's exascale computing imperative.

EXASCALE COMPUTING PROJECT

ECP Director: Doug Kothe
ECP Deputy Director: Lori Diachin

Thank you to all collaborators in the ECP and broader computational science communities. **The work discussed in this presentation represents creative contributions of many people who are passionately working toward next-generation computational science.**